Anomaly Detection from Network Traffic in IoT Devices

Deva Kumar Gajulamandyam, Tiffany Li, Jinthy Swetha Mamillapalli, Sainath Veerla
Department of Applied Data Science,
San José State University, San Jose, CA

***Abstract*—The recent growth of the Internet of Things (IoT), from home devices to industrial applications, has led to an explosion of interconnected devices generating massive amounts of data. This data offers valuable insights into network behavior and hidden patterns, but its sheer volume and complexity pose significant challenges, necessitating efficient and scalable analysis techniques. This research explores the application of machine learning algorithms to analyze IoT network traffic for hidden patterns and anomalies. To identify the optimal model for anomaly detection, algorithms including K-Nearest Neighbors, Decision Tree, Naive Bayes, and Random Forests were trained and evaluated on the UNB CIC IoT dataset. The dataset comprises a diverse range of labeled network traffic data. Among the evaluated models, Random Forest achieves the highest accuracy of 99.3%, highlighting its potential for real-world implementation.**

***Index Terms*—Machine learning, Decision Tree, Random Forest, Feature Engineering, IoT, Naive Bayes**

I. Introduction

THE global landscape is becoming closer because of the Internet of Things (IoT), which is a network extending to every object such as electronics, software , sensors, and actuators. These IoT devices which range from smart home appliances to industrial tools are becoming integral part of our lives. IoT devices are revolutionizing the way humans interact with the world. But these integration also come with cybersecurity risks. Designed with convenience and for operational efficiency,many IoT devices lack security which makes these devices vulnerable to cyberattacks. This lack of robustness, built in protection can lead to data breaches, DDoS attacks, DoS attacks, etc. Without adequate protection, sensitive personal information can be exposed which can lead to identity theft, corporate espionage and other forms of cybercrime which potentially result in significant financial and reputation damage. Enhanced security features can ensure continued functionality and reliability of the interconnected devices.

A. Problem Statement

With growing interest in data generation and storage has led to increased usage of sensors and IoT devices due to the minimal setup requirements and adaptability in various environments. However, with the rise in IoT devices, cyber threats have also grown significantly targeting individuals and organizations. The anomalies in the network can be either from devices themselves or from external entities. This is a serious concern that needs to be properly addressed. Therefore , the desired goal of the project is to leverage machine learning and deep learning algorithms to build a robust model highly capable of detecting a wide range of attacks. Advanced data analysis and machine learning techniques are used to differentiate normal and anomalous traffic patterns. It identifies the deviations from expected traffic patterns, alerting administrators to security breaches or system failures.Focusing on detecting a range of cyberattacks, the model will be adept at generalizing unseen data for consistent anomaly detection. It involves analyzing varied network traffic data, including normal and anomalous traffic.

B. Problem Background

The growth of IoT devices (29 billion by 2025) revolutionizes industries like manufacturing, agriculture, and smart cities. However, most lack built-in security, and 98% of their traffic is unencrypted, making them vulnerable to cyberattacks. This necessitates a robust solution for anomaly detection in IoT network traffic. Identifying deviations from normal behavior is crucial to safeguarding network integrity and security. These deviations can indicate security breaches, system malfunctions, or coordinated attacks. With projected security investments reaching $20.78 billion by 2027, an automated, generalizable machine learning model is needed to detect and respond to these threats.

II. Literature Survey

Research was performed to understand the workings of various models and existing solutions.

One of the research explores widely used logistic regression models, in financial risk prediction. It explores the usage of logistic regression in prediction tasks. By estimating the probability of a financial risk occurring. One of the key limitations of this approach is the low accuracy of only 75.63% which is not sufficient for effective risk management. Therefore they combined logistic regression with a Gradient Boosting Decision Tree (GBDT) model. This model combines the strengths of both models by using GBDT for feature extraction and to address non-linear relationships. Logistic regression is used for the final prediction of risk probabilities. This combination resulted in a more robust model with an accuracy of 91.25%. This paper highlights the importance of combining the statistical methods, especially in the presence of complex patterns in the dataset since logistic regression could not capture hidden patterns and interactions between variables.[1]

The paper [2] presents an innovative approach to in predicting soil erosion using the Random Search- Random Forest(RS-RF) model. This model integrates the Random Forest algorithm which is highly effective in handling complex data with random search method for optimizing the model performance. This paper also emphasizes on the advantages of the random forest algorithm such as its expertise in handling non-linear data, robustness against overfitting, and efficiency in processing large datasets. Random search, a powerful tool for navigating the hyperparameter space optimizes the random forest algorithm. By fine-tuning the model parameters, the RS0RF model showed a significant change in accuracy to around 97.4% over traditional methods which shows it potential in environment data analysis.

The paper “Naive Bayes Classification Framework Model for Optimizing Prediction of Agrotourism Products: Orange Gerga” focuses on improving the prediction of Orange Gerga production using the Naive Bayes Classification algorithm. This method was chosen because of the algorithm’s accuracy and speed in learning compared to other machine learning algorithms. The methodology used combines the Naïve Bayes algorithm with the information Gain algorithm which is useful for attribute selection to enhance predictive accuracy. This combination was tested against various sampling techniques. With linear sampling, the accuracy was almost 100 percent, it was 98.75 with shuffled sampling and 80 with linear sampling. The high level of accuracy indicates a robust framework for optimizing the prediction of agrotourism products.[3]

“KNN And Naïve Bayes Algorithms for Improving Prediction of Indonesian Film Ratings using Feature Selection Techniques ” offers a comprehensive analysis of application of K-Nearest Neighbour (KNN) and Naïve Bayes algorithms in predicting the ratings of Indonesian films. This research used CRISP-DM methodology,along with effective feature selection methods and employed advanced data mining techniques.This method follows structures and systematic exploration of data. The main is to explore the effect of feature selection on performance of two algorithms. It explores several feature selection techniques and evaluates their impact on the predictive capabilities of the KNN and Naïve Bayes algorithms. The results showed that KNN outperformed slightly. The KNN algorithm showed an accuracy of 86.65% which indicates high reliability in predicting movie ratings. Naïve Bayes algorithm shows an accuracy of 85.13% with Forward selection and without feature selection, the accuracy dropped to 71.32% demonstrating the need for feature selection for enhancing machine learning models.[4]

The paper provides a novel approach for classification of Diabetes Mellitus. It addresses the issues in medical data classification including class imbalance, missing values using SMOTE technique, Genetic Algorithm(GA) and Decision Tree(DT). It involves four layers : pre- processing, dimensionality reduction, training and performance evaluation. In the data pre-processing step, missing values, outlier detection and over sampling are dealt with. Then the dimensionality reduction and feature selection are performed reducing training complexity and to prevent overfitting. Then the decision tree algorithm is employed and the model is evaluated against metrics such as classification layer, error, precision and Area under ROC(AUROC). The accuracy of this prediction model is around 82.12 which underscores the model’s effectiveness in prediction of diabetes mellitus.[5]

III. Crisp-DM Approach

The project has been implemented using the CRISP-DM approach where the problem has been divided into phases for the complete and smooth execution of the project.

1. Business Understanding: This stage involves an in-depth understanding of the security concerns posed by IoT devices, largely focusing on network vulnerabilities. The team identified the key issue about the anomaly detection in network traffic, which could indicate potential security threats. This phase also includes setting objectives, defining the scope, and researching how anomaly detection could enhance IoT security.

2. Data Understanding: After choosing the UNB CIC IoT Dataset 2023, the team performed a detailed understanding and analysis of the data. This includes examining the structure of the dataset, the types of network traffic, and the various types of attack vectors represented. The team also conducted a thorough assessment of the data quality, and for any potential issue that may impact the model’s performance.

3. Data Preparation: The team has been concentrating on preprocessing data to make it suitable for analysis. This stage also includes adjusting the outliers using Z Score methodology, normalizing data to address the consistency concerns across different scales. To ensure data readiness these steps have been essential. 4. Data Modeling: In order to identify the best approach for anomaly detection team experimented with different Machine Learning algorithms. This involved testing and comparison of algorithms such as KNN, Naive Bayes, Decision Tree, Random Forest, and SVM. To reduce the dimensionality and focus on most informative features, which in turn optimize model performance, team implemented chi-square tests for the feature selection using PCA. 5. Data Evaluation: This stage focused on evaluating the performance of each ML model. Based on their accuracy and ability to detect anomalies, the team has assessed these models. This evaluation was aimed to identify the best results in terms of providing best accuracy and efficient functioning of anomaly detection.

6. Deployment: Streamlit was used to implement the successful model for deployment. In order to make sure that this model can be used for detecting anomalies in IoT network traffic, this application has been integrated into the real-world environment. The team primary objective has been to create an accessible UI, whilst at the same time also to ensure the stability and reliability in a live environment.

A clear representation of CRISP-DM methodology has been provided in Figure 1

IV. Architecture

The workflow in Figure 2 begins with Data collection where UNB CIC IOT Dataset 2023 is collected and stored

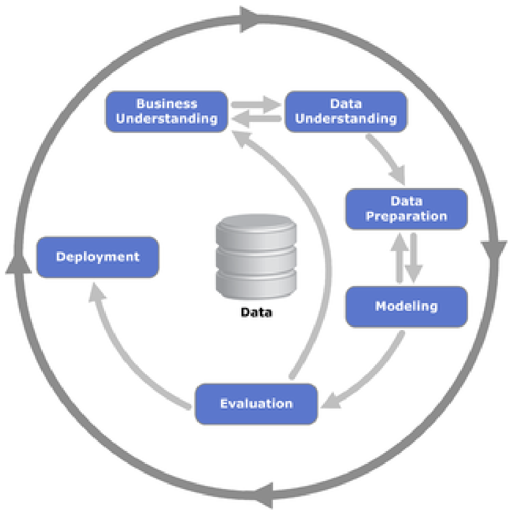


Fig. 1. CRISP DM methodology

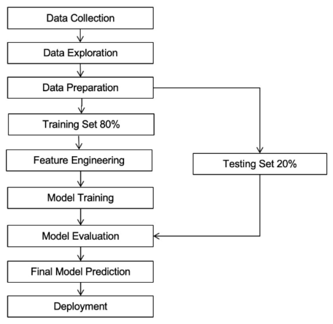


Fig. 2. Workflow

on the cloud as well in the local system as a backup. Then, Data exploration task is performed to understand the data characteristics and data distribution. Data preparation phase involves data cleaning, data engineering such Zscore outlier detection for numerical data points, range normalization, etc and splitting the dataset into training and testing. Then Feature engineering is performed on the dataset to find the important features that contribute to better prediction. After that, the model is trained followed by model evaluation involving assessment of model accuracy and predictions. Once the model is able to predict accurately the model is saved as a pickle file which consists of information of the trained model in binary format. Using Flask a UI has been created for better understanding of users. The UI unpacks the pickle file and predict on user given data.

V. Data Understanding

In this phase, exploratory data analysis of data is performed for an in-depth analysis of the data. This phase gives details a better distribution of data and an understanding of handling any uncertainty in the data.

A. Dataset

The dataset employed in this project is UNB CIC IOT Dataset 2023. It is created by University of New Brunswick

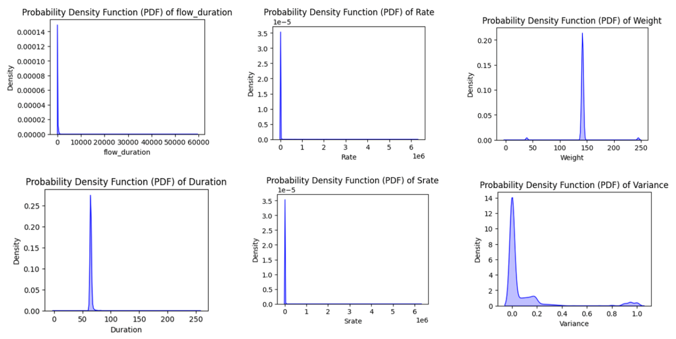


Fig. 3. Probability Distribution Graph

and Canadian Institute of Cybersecurity(CIC). Data is created by Deploying 105 real IoT devices to stimulate a smart home IoT environment. Then, 33 attacks are passed on the IoT topology, targeting IoT devices. They collected data from these attacks such as network traffic, device logs, flow_ duration and other relevant information. The data is in csv format of size 12.8GB. They are 47 features including target and 234745 records.

The dataset comprises of multiple CSV files, each of the file is programmatically read and concatenated into a single dataframe. This merged dataset forms the basis of for subsequent analysis. Upon merging , the data structure was analyzed, including the number of features, type of data in each column and basic statistical insights. Upon observation, it is found there are no null values and duplicates in our dataset which underscores the robustness of our dataset.

B. *Exploratory Data Analysis*

The next step in the CRISP -DM methodology is Exploratory Data Analysis(EDA). The main aim of this phase is to uncover hidden patterns, spot anomalies, and identify important variables through visual and statistical methods. Firstly, the distribution of various numerical features is observed. For each feature such as “flow_duration”, “Duration”, ”Rate”, etc., Probability Density Function(PDF) is plotted using kernel Density Estimation(KDE) plot. These graphs gave insights about the skewness and shape of the data distribution. A pie chart is plotted to view the proportion of attack vs benign traffic within the dataset. Figure 4 shows the pie chart. For binary features such as “fin_flag_number”, “syn_flag_number” count plots are created which helps in understanding of these flags in benign vs attack traffic. Figure 5 shows the count plots of these variables A correlation matrix is visualized using heatmap as shown in Figure 6 through which correlated features in the dataset are identified. For further exploring the numerical features, boxplots are visualized for selected features such as “Header_Length”, “Duration”,”Rate” ,etc. which is useful for identifying outliers as shown in Figure 7.

The Target feature “label” is a categorical value consisting of 23 distinct elements as in /Figure 8. The analysis aids in observing the imbalance in the dataset. Further upon refining the target feature, we mapped 23 distinct elements into 7 broad categories Later, one hot encoding is performed on this field

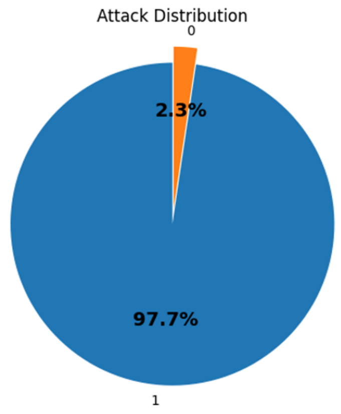


Fig. 4. Attack Distribution

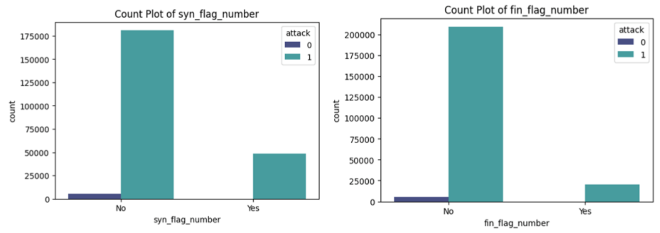


Fig. 5. Count plot of flags

converting into a more suitable format for machine learning algorithms.

C. *Data Processing*

Based on analysis in the previous phase necessary actions will be taken which include handling the imbalance and outliers. This is the subsequent phase in the project to enhance the quality and structure of the data which eventually improves model performance. As part of this step, performed operations such as outlier detection , range normalization and balancing the dataset using SMOTE.

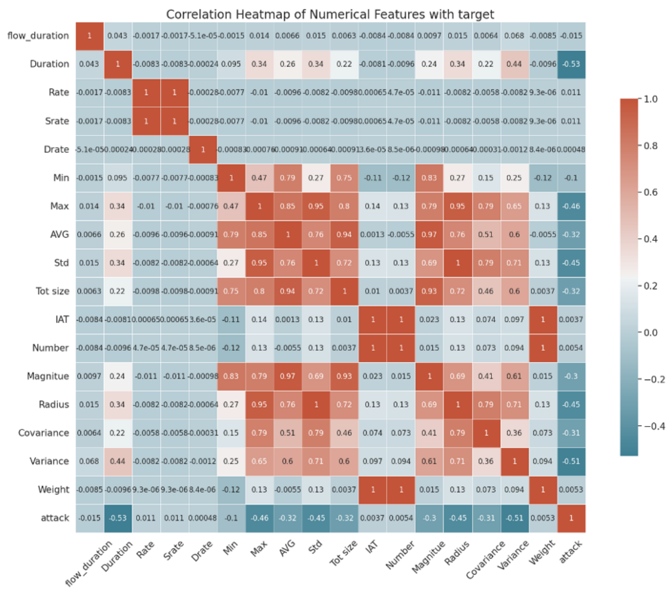


Fig. 6. Correlation Matrix

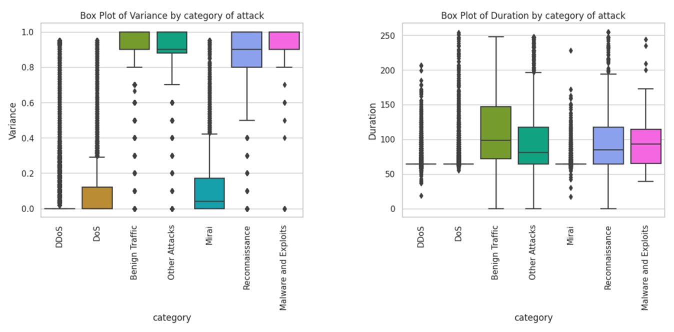


Fig. 7. Box Plots

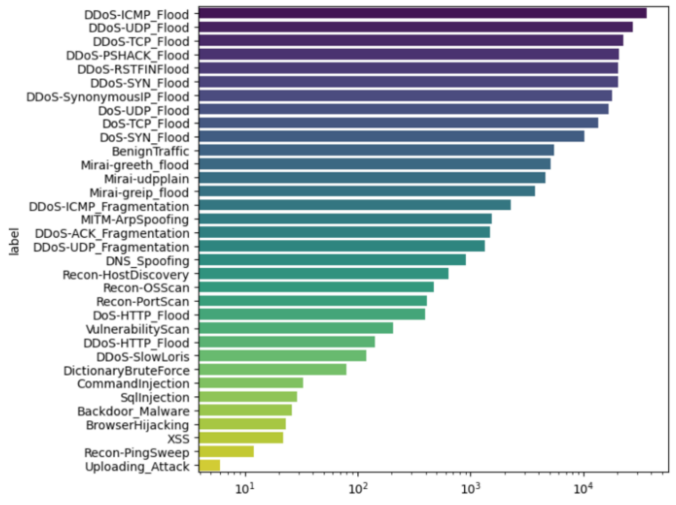


Fig. 8. Visual representation each label category

Outliers are data values which deviate significantly from the majority of the data. Presence of outliers can distort the statistical assumptions of predictive models such as changing the slope and intercept of the regression line in regression analysis which lead to incorrect predictions. For dealing with outliers which were identified during EDA , the z_score method is used. This method is a popular choice for huge datasets. Z_score is a numerical measurement describing the

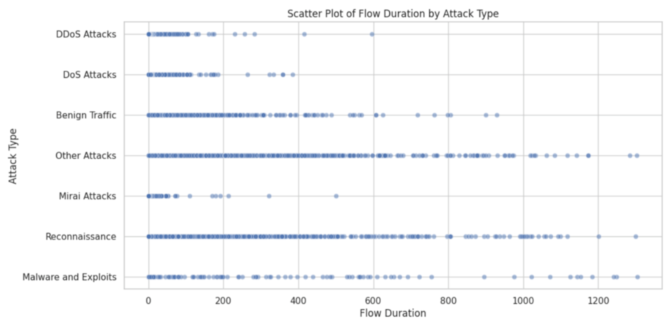


Fig. 9. *Scatter plot of flow duration by attack type after removing outliers*

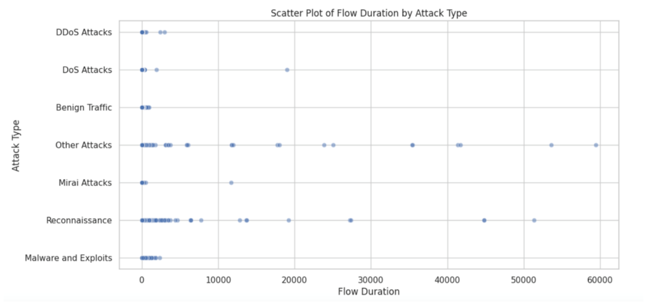
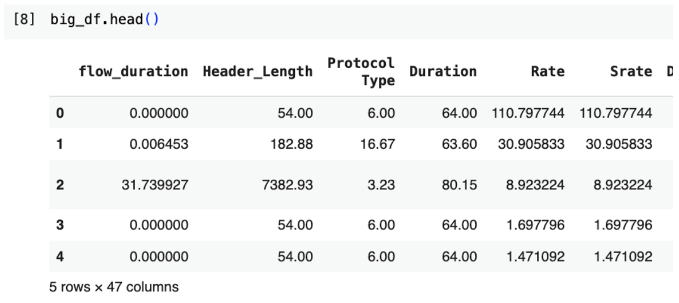


Fig. 10. *Scatter plot of Flow Duration by attack type before Removing outliers*



```
[8] big_df.head()
```

	flow_duration	Header_Length	Protocol Type	Duration	Rate	Srate
0	0.000000	54.00	6.00	64.00	110.797744	110.797744
1	0.006453	182.88	16.67	63.60	30.905833	30.905833
2	31.739927	7382.93	3.23	80.15	8.923224	8.923224
3	0.000000	54.00	6.00	64.00	1.697796	1.697796
4	0.000000	54.00	6.00	64.00	1.471092	1.471092

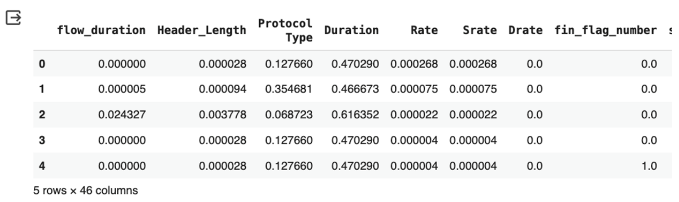
5 rows × 47 columns

Fig. 11. *Top 5 row before Normalization*

relationship between a particular data point to the mean of a group of values in terms of measurement of standard deviation from the mean. By calculating the extent the data point deviates with other groups makes the process easier to identify and filter out extreme values. A threshold of 4 is set which ensures to capture only the most extreme deviations. Calculated z_score for each data point and filtered out the values greater than 4 or less than -4. Figure 9 and Figure 10 represent scatter plots of Flow Duration before and after removal of outliers. This represents the impact of the outlier removal process.

Range normalization is a fundamental step in data processing particularly when different features in the dataset have different scales and units. This process involves scaling all the numeric features in the range between 0 and 1. Without normalization, features with larger values unduly impact the model training process resulting in biased results. To counteract this issue range normalization using "MinMaxScaler" from Scikit-Learn is employed. It is a popular choice for normalization particularly useful in algorithms which rely on distance calculation such as K-Nearest Neighbours. It ensures that each feature contributes equally to final prediction without any bias. Values of numeric columns before and after applying normalization are shown in Figure 11 and Figure 12.It can be deduced that all the values are in a specified range after min-maxing.

In this project, it is of paramount importance to deal with an imbalance dataset for the development of a reliable predictive model. Imbalance in the dataset can lead a model to perform well on the majority class but poorly on the minority class. In this case, the accuracy in detection of rare events becomes less. To mitigate this problem, SMOTE technique is used which balances the dataset by augmenting the minority class. This technique creates new samples which are similar but not identical to existing ones, thereby enriching the diversity in the dataset. Before application of this method, the dataset is split into training and testing sets. After that, SMOTE is performed only on the training set. This is done to prevent introducing synthetic data into the testing set which may lead to overoptimistic results .For demonstration of SMOTE impact, the class distribution before and after SMOTE is analyzed. Figure 13 shows before and after balancing results. It can be seen that after sampling , all the sizes of each category are the same.



	flow_duration	Header_Length	Protocol Type	Duration	Rate	Srate	Drate	fin_flag_number
0	0.000000	0.000028	0.127660	0.470290	0.000268	0.000268	0.0	0.0
1	0.000005	0.000094	0.354681	0.466673	0.000075	0.000075	0.0	0.0
2	0.024327	0.003778	0.068723	0.616352	0.000022	0.000022	0.0	0.0
3	0.000000	0.000028	0.127660	0.470290	0.000004	0.000004	0.0	0.0
4	0.000000	0.000028	0.127660	0.470290	0.000004	0.000004	0.0	1.0

5 rows × 46 columns

Fig. 12. *Top 5 rows after Normalization*

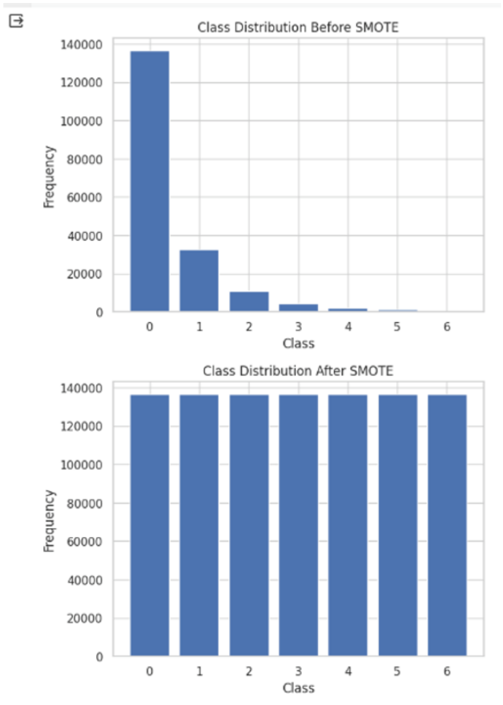


Fig. 13. Target feature distribution before and after balancing

VI. Feature Engineering

There are 43 features in the dataset, therefore feature engineering has to be performed which involves selecting, modifying and creating features that significantly improves the model performance. Advanced techniques like Principal Component Analysis (PCA) , Random Forest Classifier for feature importance and the Chi- Squared method are used to refine the feature set.

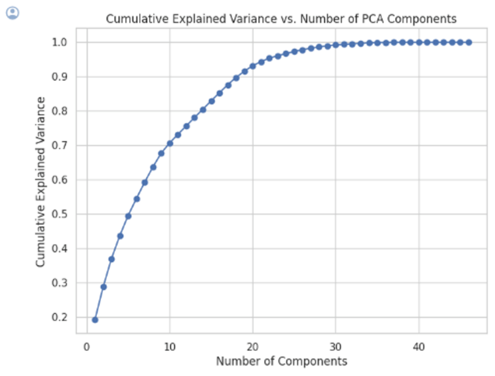
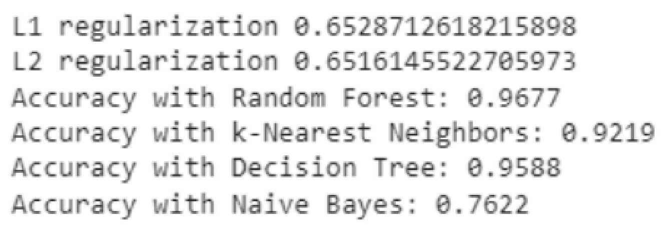


Fig. 14. *PCA Plot*



```
L1 regularization 0.6528712618215898
L2 regularization 0.6516145522705973
Accuracy with Random Forest: 0.9677
Accuracy with k-Nearest Neighbors: 0.9219
Accuracy with Decision Tree: 0.9588
Accuracy with Naive Bayes: 0.7622
```

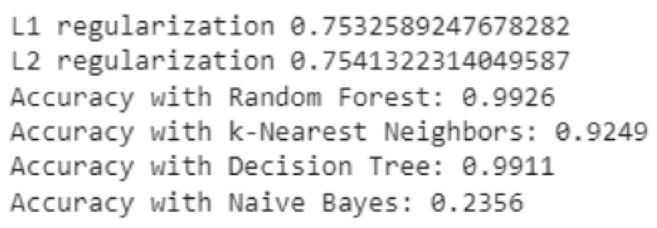
Fig. 15. PCA result

A. *Principal Component Analysis(PCA)*

PCA is one the popular dimensionality reduction techniques in datasets with a high dimensionality, where each data point is represented by various features. It transforms the features into new sets of uncorrelated features(principal components) which capture most of the variance. This technique helps reduce complexity without losing essential information, critical for computational efficiency. Before applying PCA, "StandardScaler " was utilized which ensures that each feature equally contributes to the analysis. Then, PCA was applied to determine the components that retain almost 95% of the variance in the dataset. This threshold was chosen as a tradeoff between reducing dimensionality and preserving information. PCA approach not only helps in reducing the dimensionality in the dataset but also in mitigating multicollinearity which occurs when the features are highly correlated to each other. Figure 14 illustrates the results of the PCA performed. It shows the plot of cumulative variance against the optimal number of components. From this visualization, it can be determined the 22 components to be included in the training the model to explain 95% variance in the dataset.

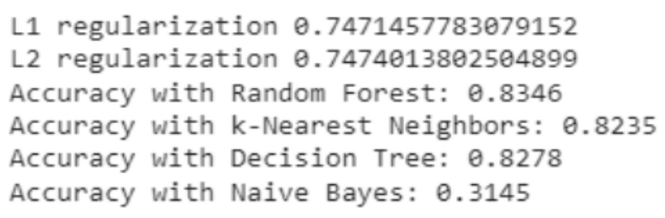
B. *Random Forest Classifier for Feature Importance*

The Random Forest Classifier is not only a powerful classifier but also an effective tool for feature selection. In this method, during the construction of the multitude of decision trees , the classifier determines how each feature can contribute to the purity of the split at every node . This is then evaluated which key metrics such as Gini impurity or information gain. Features that consistently result in more effective splits which eventually result in an increase in purity of the nodes



```
L1 regularization 0.7532589247678282
L2 regularization 0.7541322314049587
Accuracy with Random Forest: 0.9926
Accuracy with k-Nearest Neighbors: 0.9249
Accuracy with Decision Tree: 0.9911
Accuracy with Naive Bayes: 0.2356
```

Fig. 16. Feature Importance



```
L1 regularization 0.7471457783079152
L2 regularization 0.7474013802504899
Accuracy with Random Forest: 0.8346
Accuracy with k-Nearest Neighbors: 0.8235
Accuracy with Decision Tree: 0.8278
Accuracy with Naive Bayes: 0.3145
```

Fig. 17. Chi square

are assessed as important. This approach is particularly useful when there are too many features in the dataset, like this case. By focusing on fewer and important features, complexity in the model can be reduced and interpretability can be enhanced. Therefore, Random Forest classifier was allowed to train on the dataset and obtain the list of features by ranking based on their importance scores suggesting their influence on the model in predicting the outcome. Concentrating on fewer features ensures the model is manageable.

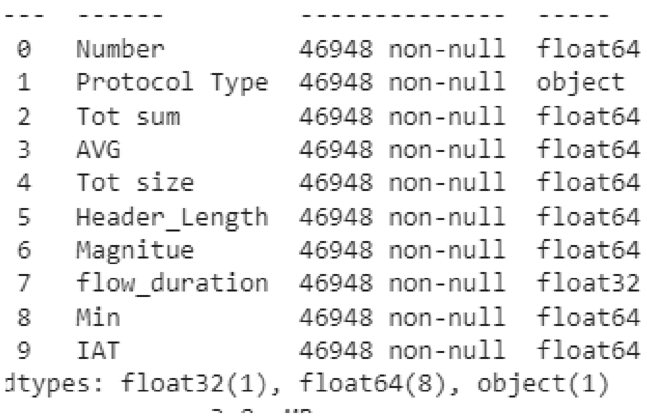
C. *Chi- Squared method*

The Chi- Squared method which is used in statistical hypothesis testing is widely used in feature selection, especially for categorical variables in machine learning. It is based on the chi-squared statistics which evaluates independence between features. It determines if the relation between each feature and target feature is by chance or is it statistically significant by comparing the frequencies observed in the data and expected frequencies if the variables are independent . Variables which exhibited higher Chi -square value can be interpreted as having significant association with the target variable while the low chi-square values as weakly associated. This method is utilized for the dataset and for each feature chi-square statistic was found with respect to the target variable. Features with higher value were retained as these features which increase model accuracy and avoid overfitting.

The 43 features have been reduced to 10 features as in Figure 18using the random feature importance method that have been shown in deployment. As feature importance using random forest better performance when tested on based models.

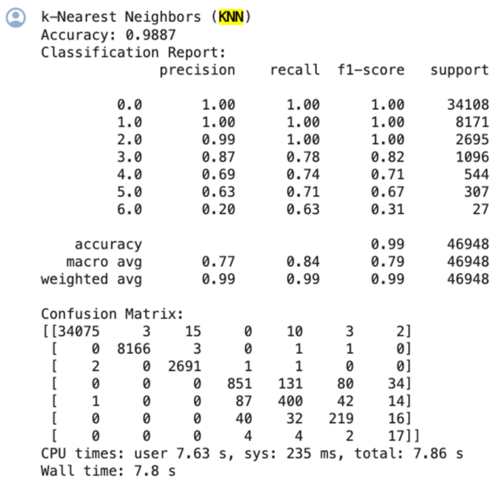
VII. Model Selection

In this section, different models are trained to check their performance metrics and their ability to predict data. Based on analysis from feature engineering only the selected features are used instead of all the features for modeling purposes there



```
---  ------          -------------  -----
 0   Number          46948 non-null  float64
 1   Protocol Type   46948 non-null  object
 2   Tot sum         46948 non-null  float64
 3   AVG             46948 non-null  float64
 4   Tot size        46948 non-null  float64
 5   Header_Length   46948 non-null  float64
 6   Magnitue        46948 non-null  float64
 7   flow_duration   46948 non-null  float32
 8   Min             46948 non-null  float64
 9   IAT             46948 non-null  float64
dtypes: float32(1), float64(8), object(1)
```

Fig. 18. selected columns



```
k-Nearest Neighbors (KNN)
Accuracy: 0.9887
Classification Report:
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00     34108
         1.0       1.00      1.00      1.00      8171
         2.0       0.99      1.00      1.00      2695
         3.0       0.87      0.78      0.82      1096
         4.0       0.69      0.74      0.71       544
         5.0       0.63      0.71      0.67       307
         6.0       0.20      0.63      0.31        27

    accuracy                           0.99     46948
   macro avg       0.77      0.84      0.79     46948
weighted avg       0.99      0.99      0.99     46948

Confusion Matrix:
[[34075     3    15     0    10     3     2]
 [    0  8166     3     0     1     1     0]
 [    2     0  2691     1     1     0     0]
 [    0     0     0   851   131    80    34]
 [    1     0     0    87   400    42    14]
 [    0     0     0    40    32   219    16]
 [    0     0     0     4     4     2    17]]
CPU times: user 7.63 s, sys: 235 ms, total: 7.86 s
Wall time: 7.8 s
```

Fig. 19. Classification Report for KNN

by reducing the data that is to be trained.

A. KNN

KNN is a powerful algorithm used in predictive modeling, particularly in classification problems. It is a non-parametric algorithm that does not make any assumptions based on data distribution which makes it suitable for real world scenarios. Unlike many other algorithms , KNN does not require a separate training phase and it is relatively easy to implement.

It works by identifying the nearest 'K' data values to the target point based on Euclidean distance. The concept of this algorithm is that similar data points likely belong to the same class. This approach assigns the data point to a class on the basis of the majority of its nearest neighbors. When a value needs to be assigned to a class, the distance between the point and all the other points is calculated. Then the distances are sorted and 'k' nearest points are identified. Then, the algorithm searches at the classes of these 'k' points and assigns the class based on the most frequent one. The most important factor is the choice of 'k' value. Smaller 'k' value makes the algorithm sensitive to noise while a larger one makes it computationally demanding and less accurate.

For effective implementation of KNN, the data must be scaled, missing values must be handled and one hot encoding must be done on categorical variables. All these requirements are handled in the data preparation phase of the project. KNN is implemented on the dataset and the classification report is shown in figure 19.

The model has achieved an accuracy of 98.87%. The classification provides in depth insights about the performance metrics for every class. The model shows a perfect precision and recall scores for classes 0 and 1 which are 'DDOS Attacks' and 'DOS Attacks' respectively which is an indication that the model is highly sensitive for these classes. Highest F1 scores are achieved for these classes.

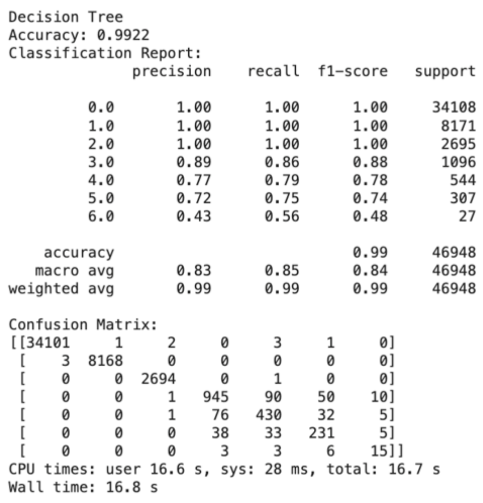
B. Decision tree

Decision Trees are a type of supervised learning algorithm used for both classification and regression tasks. One of the significant advantages of decision trees is their interpretability. They can also handle both numeric and categorical data effectively without need for transformations. They can model non-linear relationships between features and target variables which is helpful in complex datasets . This method is also robust to outliers and has faster training time. All these properties make it a justifiable choice for this project with its complexity in the dataset and non-linear relationship between features. A decision tree consists of nodes, branches and leaves. The decision making process starts at the root node and splits the data on the feature which gives the largest information gain or Gini impurity. This is continuously performed until it reaches a maximum depth or has few instances to split.One of the disadvantages of this model is its tendency to overfit. Ensemble techniques used be to mitigate the issue.

The hyperparameters such max_depth are tuned properly to ensure better performance of the model. The classification report generated after applying the Decision Tree is shown in figure 20. This model attained an accuracy of 99.2%. The model demonstrated high level precisions for all the classes except class 6 indicating misclassification. The recall and F1 scores are also perfect for classes 0,1,2. The rest of the classes show less recall values compared to others.

C. Naive Bayes

Naïve Bayes classifiers can be considered as a family of "probabilistic classifiers" which work on bayes theorem with strong assumptions such as conditional independence between the features. This model is highly recommended for text classification and can outperform many sophisticated classification models despite its simplicity. Naïve Bayes can handle multi class problems and is also known for effectiveness with high dimensionality data . In this implementation, involving categorizing data into multiple classes ,it is important to have such an algorithm which easily adapts to more than two outcomes. The Naïve bayes classifier will calculate the posterior probability for each class and the class with highest probability is the final outcome. The major drawback of the Naïve bayes is its assumption that all features are mutually independent which may not be the case in some cases.



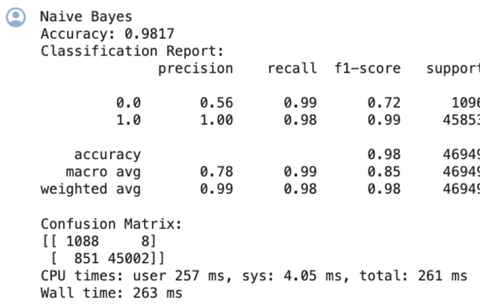
```
Decision Tree
Accuracy: 0.9922
Classification Report:
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00     34108
         1.0       1.00      1.00      1.00      8171
         2.0       1.00      1.00      1.00      2695
         3.0       0.89      0.86      0.88      1096
         4.0       0.77      0.79      0.78       544
         5.0       0.72      0.75      0.74       307
         6.0       0.43      0.56      0.48        27

    accuracy                           0.99     46948
   macro avg       0.83      0.85      0.84     46948
weighted avg       0.99      0.99      0.99     46948

Confusion Matrix:
[[34101     1     2     0     3     1     0]
 [    3  8168     0     0     0     0     0]
 [    0     0  2694     0     1     0     0]
 [    0     0     1   945    90    50    10]
 [    0     0     1    76   430    32     5]
 [    0     0     0    38    33   231     5]
 [    0     0     0     3     3     6    15]]
CPU times: user 16.6 s, sys: 28 ms, total: 16.7 s
Wall time: 16.8 s
```

Fig. 20. Classification Report of Decision Tree



```
Naive Bayes
Accuracy: 0.9817
Classification Report:
              precision    recall  f1-score   support

         0.0       0.56      0.99      0.72      1096
         1.0       1.00      0.98      0.99     45853

    accuracy                           0.98     46949
   macro avg       0.78      0.99      0.85     46949
weighted avg       0.99      0.98      0.98     46949

Confusion Matrix:
[[ 1088     8]
 [  851 45002]]
CPU times: user 257 ms, sys: 4.05 ms, total: 261 ms
Wall time: 263 ms
```

Fig. 21. Classification Report for Naive Bayes

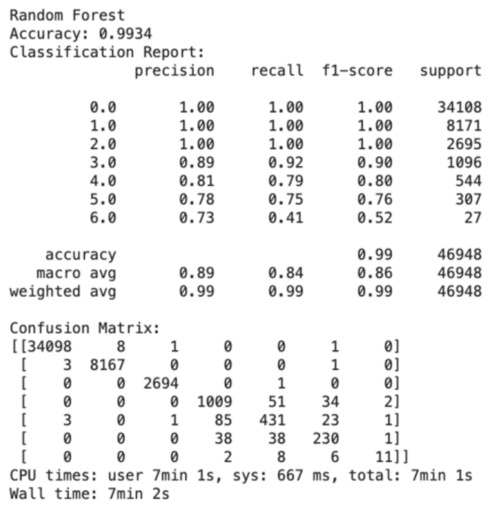
The classification report for the Naïve Bayes classifier is shown in Figure 21. Overall accuracy of the model is 98.17%. The model has a precision of 0.56 for class 0 which suggests there are many misclassification instances. The recall scores are very high for class 0 and class 1 around “0.99” and “0.98” respectively. The F1 score is highest for class 1 compared to all other classes.

D. Random Forest

Random Forest is a versatile and robust ensemble machine learning algorithm which has high accuracy and adaptability in various domains. When dealing with large datasets with higher dimensionality , random forest can maintain high accuracy even when majority data is missing.

It is less vulnerable to overfitting compared to decision trees. This algorithm merges the predictions of multiple decision trees which result in a more accurate and stable model. In this method, the Random forest uses “bagging” to create multiple decision trees. Different subsets of the data are randomly selected to train the trees. When a new data comes, each tree in the forest will predict and outcome would be the most frequent class by the tree.

The model was trained on the dataset using the method “bagging” which promotes model variance and accuracy. It is then evaluated using the test dataset and classification report is generated as shown in Figure 22. The model illustrated an



```
Random Forest
Accuracy: 0.9934
Classification Report:
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00     34108
         1.0       1.00      1.00      1.00      8171
         2.0       1.00      1.00      1.00      2695
         3.0       0.89      0.92      0.90      1096
         4.0       0.81      0.79      0.80       544
         5.0       0.78      0.75      0.76       307
         6.0       0.73      0.41      0.52        27

    accuracy                           0.99     46948
   macro avg       0.89      0.84      0.86     46948
weighted avg       0.99      0.99      0.99     46948

Confusion Matrix:
[[34098     8     1     0     0     1     0]
 [    3  8167     0     0     0     1     0]
 [    0     0  2694     0     1     0     0]
 [    0     0     0  1009    51    34     2]
 [    3     0     1    85   431    23     1]
 [    0     0     0    38    38   230     1]
 [    0     0     0     2     8     6    11]]
CPU times: user 7min 1s, sys: 667 ms, total: 7min 1s
Wall time: 7min 2s
```

Fig. 22. Classification Report of Random Forest

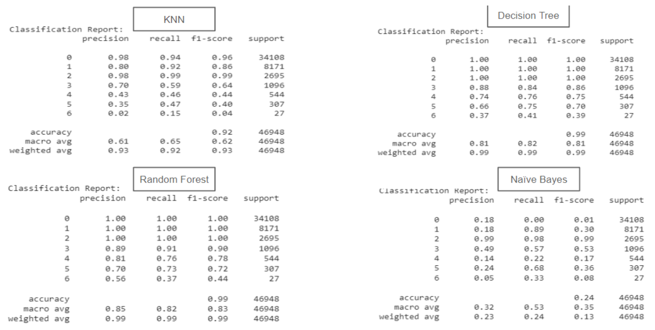
exceptional accuracy of 99.34%.The models precision ,recall and F1 scores are all perfect scores for classes 0,1,2 which are “DDoS attacks”,”DoS attacks”,”Mirai Attacks” respectively. For class 6 the scores are notably low compared to others.

VIII. Results

In comparing all four machine learning models-KNN, Decision Tree(DT), Naïve Bayes and Random Forest across various key metrics as shown in Figure 23, it can be deduced that each model has its own strengths. Random Forest has the highest accuracy with 0.993 followed by decision tree , KNN and last being Naïve Bayes. In case of precision, all the models have the same values of 0.99 ensuring minimized false positives. In terms of recall and F1 score, KNN, Random Forest and DT exhibit top performance with scores 0.99 while it is slightly lower for Naïve Bayes which has 0.98. Overall, Random Forest emerges as the most robust option as it excels in accuracy. When it comes to multiclass classification the model should not have any bias and should be able to predict all classes equally. The classification report clearly shows that Random Forest was able to predict more accurately even for less support data. This indicates that this model is more suitable for real-time analysis as it can capture the details in the dataset more precisely. Naive Bayes is not able to understand the relationship between features and the target and provides the lowest performance. As the dataset is large K-fold cross-validation has not been used a metric for testing due to its high computational resources.Therefore Random Forest is selected as the model for this project and the deployment is done using this model.

IX. Deployment

In order to deploy the best performing model and test user input, a user interface is required and this is achieved using Streamlit. Streamlit is a powerful Python library that simplifies the process of building and deploying machine learning models and data applications. Although there are multiple viable


KNN
Decision Tree
Random Forest
Naïve Bayes

```
Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.94      0.96     34108
           1       0.80      0.92      0.86      8171
           2       0.98      0.99      0.99      2695
           3       0.70      0.59      0.64      1096
           4       0.43      0.46      0.44       544
           5       0.35      0.47      0.40       307
           6       0.02      0.15      0.04        27

    accuracy                           0.92     46948
   macro avg       0.61      0.65      0.62     46948
weighted avg       0.93      0.92      0.93     46948
```

```
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     34108
           1       1.00      1.00      1.00      8171
           2       1.00      1.00      1.00      2695
           3       0.88      0.84      0.86      1096
           4       0.74      0.76      0.75       544
           5       0.66      0.75      0.70       307
           6       0.37      0.41      0.39        27

    accuracy                           0.99     46948
   macro avg       0.81      0.82      0.81     46948
weighted avg       0.99      0.99      0.99     46948
```

```
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     34108
           1       1.00      1.00      1.00      8171
           2       1.00      1.00      1.00      2695
           3       0.89      0.91      0.90      1096
           4       0.81      0.76      0.78       544
           5       0.70      0.73      0.72       307
           6       0.56      0.37      0.44        27

    accuracy                           0.99     46948
   macro avg       0.85      0.82      0.83     46948
weighted avg       0.99      0.99      0.99     46948
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.18      0.00      0.01     34108
           1       0.18      0.89      0.30      8171
           2       0.99      0.98      0.99      2695
           3       0.49      0.57      0.53      1096
           4       0.14      0.22      0.17       544
           5       0.24      0.68      0.36       307
           6       0.05      0.33      0.08        27

    accuracy                           0.24     46948
   macro avg       0.32      0.53      0.35     46948
weighted avg       0.23      0.24      0.13     46948
```

Fig. 23. Results

options for this task like Flask and Django, Streamlit is preferred as it provides a range of widgets and components that enable seamless integration of data visualizations, interactive charts, and user inputs, fostering a dynamic and engaging user experience. It's also quick and easy to develop.

For deployment, the random forest model trained on the top 10 features according to random forest feature importance is considered and saved after training as it has exhibited the best performance. This saved model is used in the deployment process to make predictions on unseen data. A simple web interface with accessibility to insert values for these 10 features is created. After entering the values, the submit button is used to collect this data and send it to the saved model for prediction. This prediction is converted back to corresponding class name (attack type) and displayed on the screen. A sample prediction and it's output is shown in Fig. 24 and Fig. 25.

A. *Source code*

GitHub Repo Link: https://github.com/gdevakumar/anomaly-detection-iot-networks

B. *Presentation*

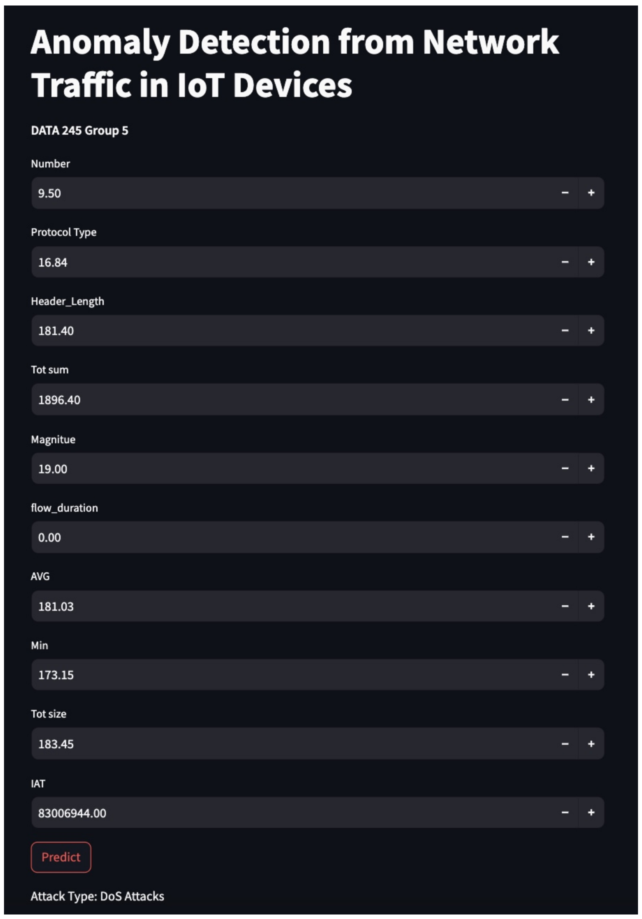
Presentation Link: https://www.youtube.com/watch?v=AFHqHWSNTzw

C. *Dataset*

Dataset Link: https://drive.google.com/drive/folders/1SehzpbQe8P1rjNfsekyYaZIqSWq1OIJg?usp=sharing

X. Conclusion

The projects focused on improving the security of IoT networks by leveraging machine learning model for anomaly detection. Four algorithms, KNN, Decision Tree, Naive Bayes, and Random Forest, were evaluated on the UNB CIC IoT dataset after effective data preparation. Out of the all models implemented, the Random Forest model achieved the highest accuracy of 99.3% with commendable precision and F1 scores. This demonstrated the effectiveness of machine learning in safeguarding IoT networks by pinpointing and mitigating potential cyber threats which presents the potential of machine learning models for anomaly detection in IoT networks. With increased security , there would be significant increase in market for IoT devices as the utilization expands.



Anomaly Detection from Network Traffic in IoT Devices

DATA 245 Group 5

Number: 9.50

Protocol Type: 16.84

Header_Length: 181.40

Tot sum: 1896.40

Magnitue: 19.00

flow_duration: 0.00

AVG: 181.03

Min: 173.15

Tot size: 183.45

IAT: 83006944.00

Attack Type: DoS Attacks

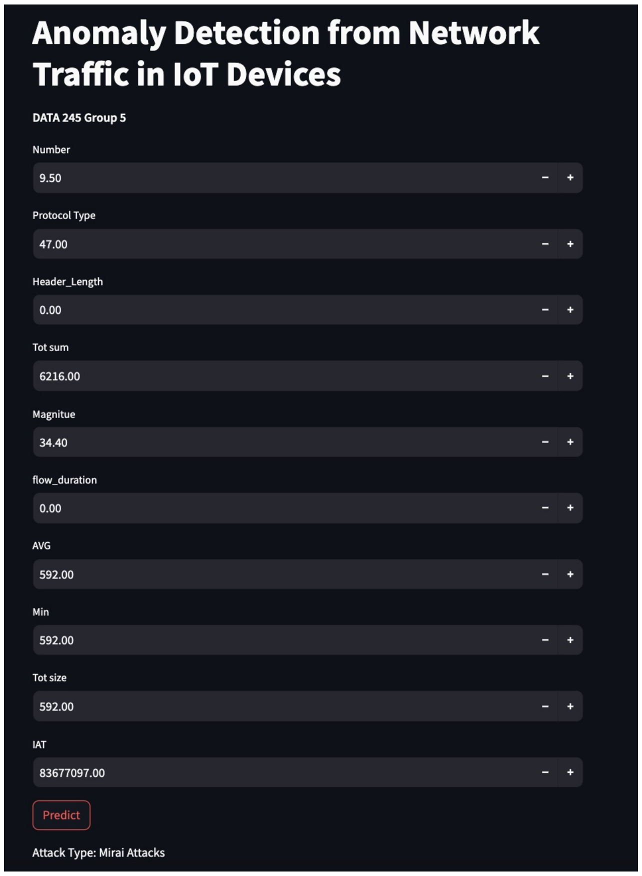
Fig. 24. Sample input and prediction - 1

XI. Limitations

Though the model exhibits admirable accuracy showcasing it potential in anomaly detection in IoT networks, there are still some drawbacks. One of the major limitation of the project is limited generalization because of the dataset scope. Diverse IoT devices can be combined which would enhance the model to become more robust and can be applicable to larger scale networks. Another limitations is lack of real time processing by the model which is crucial for timely anomaly detection in live IoT environments. Therefore, Integrating data in real time would significantly increase the usability of the model.

XII. Future Scope

The project aims for better collection of data making the model generalised for different data collected. In the project the model has been particularly trained on IoT data sets. When considering a real world situation nothing an be certain and the model should be able to predict the new unseen data accurately. This can enhanced by training the data on more classes and raw data that is directly received to the systems. By focusing on these prospects we intend to create an enhance model that aids in live detection of networks.



Anomaly Detection from Network Traffic in IoT Devices

DATA 245 Group 5

Number
9.50

Protocol Type
47.00

Header_Length
0.00

Tot sum
6216.00

Magnitue
34.40

flow_duration
0.00

AVG
592.00

Min
592.00

Tot size
592.00

IAT
83677097.00

Attack Type: Mirai Attacks

Fig. 25. Sample input and prediction - 2

References

[1] M. Duan, “Analysis of corporate financial risk avoidance strategies based on logistic regression model,” *Applied Mathematics and Nonlinear Sciences*.

[2] Z. Tarek, A. M. Elshewey, S. M. Shohieb, A. M. Elhady, N. E. El-Attar, S. Elseuofi, and M. Y. Shams, “Soil erosion status prediction using a novel random forest model optimized by random search method,” *Sustainability*, vol. 15, no. 9, p. 7114, 2023.

[3] Y. Isro’Mukti, V. Itteridi, I. Sulaini, and Y. Widiastiwi, “Naive bayes classification framework model for optimizing prediction of agrotourism products orange gerga,” in *2022 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS)*. IEEE, 2022, pp. 343–347.

[4] J. Wiratama, R. S. Oetama, *et al.*, “Knn and naïve bayes algorithms for improving prediction of indonesian film ratings using feature selection techniques,” in *2023 4th International Conference on Big Data Analytics and Practices (IBDAP)*. IEEE, 2023, pp. 1–6.

[5] C. Azad, B. Bhushan, R. Sharma, A. Shankar, K. K. Singh, and A. Khamparia, “Prediction model using smote, genetic algorithm and decision tree (pmsgd) for classification of diabetes mellitus,” *Multimedia Systems*, pp. 1–19, 2021.